Coding Challenge

Include a link to the hosted repository (e.g. Github, Bitbucket...). We cannot review archives or single files. The repo should include a README that follows the principles described below In particular, please make sure to include high-level explanation about what the code is doing. Ideally, the code you're providing:

- Has been written by you alone. If not, please tell us which part you wrote and are most proud of in the README.
- Is leveraging web technologies.
- Is deployed and hosted somewhere.

Readme

Write your README as if it was for a production service. Include the following items:

- Description of the problem and solution.
- Reasoning behind your technical choices, including architectural. Trade-offs you might have made, anything you left out, or what you might do differently if you were to spend additional time on the project.
- Link to other code you're particularly proud of.
- Link to your resume or public profile.
- Link to to the hosted application where applicable.

How we review

The aspects of your code we will judge include:

- Clarity: does the README clearly explains the problem and solution?
- Correctness: does the application do what was asked? If there is anything missing, does the README explain why it is missing?
- Code quality: is the code simple, easy to understand, and maintainable? Are there any code smells or other red flags?
- Testing: how thorough are the automated tests? Will they be difficult to change if the requirements of the application were to change?
- UX: is the web interface understandable and pleasing to use?
- Technical choices: do choices of libraries, databases, architecture etc. seem appropriate for the chosen application?

Problem Statements

Please organize, design, test, document and deploy your code as if it were going into production, then send us a link to the hosted repository (e.g. Github, Bitbucket...).

Functional spec

Prototype one of the following projects:

- Departure Times
- SF Movies
- Email Service
- Food Trucks

The UX/UI is totally up to you. If you like, get creative and add additional features a user might find useful!

Departure Times

Create a service that gives real-time departure time for public transportation (use freely available public API). The app should geolocalize the user.

Here are some examples of freely available data:

- 511 (San Francisco)
- Nextbus (San Francisco)

SF Movies

Create a service that shows on a map where movies have been filmed in San Francisco. The user should be able to filter the view using autocompletion search.

The data is available on DataSF: Film Locations.

Email Service

Create a service that accepts the necessary information and sends emails. It should provide an abstraction between two different email service providers. If one of the services goes down, your service can quickly failover to a different provider without affecting your customers.

Example Email Providers:

- SendGrid
- Mailgun
- Mandrill
- Amazon SES

All listed services are free to try and are pretty painless to sign up for, so please register your own test accounts on each.

Food Trucks

Create a service that tells the user what types of food trucks might be found near a specific location on a map.

The data is available on DataSF: Food Trucks

Technical spec

The architecture will be split between a back-end and a web front-end, for instance providing a JSON in/out RESTful API. Feel free to use any other technologies provided that the general client/service architecture is respected. Include both frontend and backend.

Back-end

We believe there is no one-size-fits-all technology. Good engineering is about using the right tool for the right job, and constantly learning about them. Therefore, feel free to mention in your README how much experience you have with the technical stack you choose, we will take note of that when reviewing your challenge.

Here are some technologies we are more familiar with:

Python (most of our back-end uses this language)
JavaScript
Ruby
PHP
Go
Java

You are also free to use any web framework. If you choose to use a framework that results in boilerplate code in the repository, please detail in your README which code was written by you (as opposed to generated code).

Front-end

The front-end should ideally be a single page app with a single index.html linking to external JS/CSS/etc. You may take this opportunity to demonstrate your CSS3 or HTML5 knowledge. Using a frontend MVC framework is recommended.